Fault Tolerance in RNS: An Efficient Approach

Damu Radhakrishnan and Taejin Pyon

Department of Electrical Engineering
University of Idaho, Moscow, ID 83843

Abstract

The main objective of this paper is to minimize the error correction hardware for single fault tolerance in Residue Number Systems (RNS). With this objective, a new approach for the design of an error calculator for single fault tolerance in RNS arithmetic is presented. It corrects the error concurrently during normal operation. Furthermore, the computation of each error is achieved in parallel by a number of smaller look-up tables. This brings forth a considerable reduction in the total error calculator hardware.

1 Introduction

The RNS is receiving much attention recently because of their ability to do arithmetic on a number of independent and parallel processors without any carry propagation. In general, a large integer can be subdivided into a set of smaller integers, with the arithmetic operations performed concurrently on them. This parallelism increases the overall arithmetic computation speed. Moreover, it provides a basis for the design of fault tolerant hardware architectures [1,2].

Cosenzino [3] from MITRE Corp. designed a single fault tolerant RNS FIR filter using a look-up table error calculator proposed originally by Watson [4]. In his design, a PLA was used to implement the error calculator, but, in general, this error calculator may consume a large portion of the chip area especially for large RNS-based systems. This paper, on the other hand, presents the design of a highly compact error calculator.

2 Residue Number System

In RNS, an integer X is uniquely represented by a l-tuple of integers \((x_1, x_2, ..., x_l)\), called the residue representation of \(X\) [2]. The integers \(x_i, i = 1, 2, ..., l\) are the residues and are obtained as remainders when the number \(X\) is divided by a set of distinct, and relatively prime integers \(m_i, i = 1, 2, ..., l\). These integers are called the moduli. Thus \(x_i = X \mod m_i\), and is denoted by \([X]_{m_i}\), where \(0 \leq x_i < m_i\). It follows from the Chinese Remainder Theorem [2] that for any given l-tuple, \((x_1, x_2, ..., x_l)\), satisfying the above relationships, there exists one and only one integer \(X\) such that \(0 \leq X < \Pi_{i=1}^{l} m_i\). The interval \([0, M]\) with \(M = \Pi_{i=1}^{l} m_i\) will be called the legitimate range and represents the useful computational range of the number system.

Fault tolerance in RNS can be achieved by adding \(r (r > 2)\) additional moduli, forming a redundant residue number system (RRNS). \(m_1, m_2, ..., m_r\) are called non-redundant moduli, and \(m_{r+1}, m_{r+2}, ..., m_{l+r}\) are called the redundant moduli. A number \(X\) in RRNS is represented by \(l + r\) residue digits \((x_1, x_2, ..., x_{r+l})\). \((x_1, x_2, ..., x_l)\) are the non-redundant residue digits, and \((x_{r+1}, x_{r+2}, ..., x_{l+r})\) are the redundant digits. The total number range is \([0, M_F]\), where \(M_F = \Pi_{i=1}^{l+r} m_i\). The interval \([M, M_F]\) is called the illegitimate range.

3 Overview of Previous Work

Two different approaches to fault tolerant realizations exist in RRNS. One approach is based on the theory of projections [5]. A pipelined self-checking error checker has been developed in [6,7]. This error checker uses a Mixed Radix Converter (MRC). Fault detection is accomplished by checking for non-zero redundant mixed radix digits. Once a fault is detected, a control circuit performs the correction.

A second approach is based on the reconstruction of the redundant residue digits by base extension [3,4]. In the mid 1960's Watson [4] reported a single digit error correction scheme using base extension and a single look-up table. To illustrate his scheme, let us consider an RRNS with \(l + 2\) moduli. Let \(X = (x_1, x_2, ..., x_l, x_{l+2})\) represent the number \(X\) with a fault in its \(l^{th}\) digit. First, the redundant digits \(x_{l+2}\) and \(x_{l+3}\) are derived from \((x_1, x_2, ..., x_l, x_{l+2})\) by base extension, and then the error values \(x_{l+1} = \lfloor x_{l+1} - x_{l+3} m_i \rfloor_{m_i}\), and \(x_{l+2} = \lfloor x_{l+2} - x_{l+3} m_i \rfloor_{m_i}\) are calculated. If both values are zeros, then no error has occurred. If one of them is non-zero, then either \(x_{l+1}\) or \(x_{l+2}\) has an error. If both are non-zero, then the error is in one of the non-redundant residues. Figure 1 shows a concurrent error correction scheme based on the above [3] using a set of smaller moduli. The error values \(x_{l+1}\) and \(x_{l+2}\) are used to address an error calculator look-up table. A PLA was used in [3] for implementing the look-up table. However, in large RNS-based systems, the look-up table will become quite large, making it impractical. Hence, a new approach for a single digit error correction is presented here which reduces the size of the error calculator significantly.

4 New Fault Tolerant Design

This new error calculator does not depend on a single look-up table, but instead, the computation of each error is achieved independently by each redundant moduli operating in parallel. This concept is based on base extension [3,4]. Separating the calculation into individual redundant moduli in this manner
Consider a number \(X\), represented by \(n\) non-redundant residue digits \((x_1, x_2, \ldots, x_n)\). If a single fault occurs on the \(i\)th residue digit \((x_i)\) of \(X\), then this faulty number is denoted as \(\bar{X}_i = (x_1, x_2, \ldots, \bar{x}_i, \ldots, x_n)\). The erroneous number \(\bar{X}_i\) can now be denoted as [3]:

\[
\bar{X}_i = (x_1, \bar{x}_2, \ldots, \bar{x}_i, \ldots, x_n)
\]

\[
= (x_1, x_2, \ldots, x_i, \bar{x}_i, \ldots, x_n) + (0, 0, \ldots, e_i, 0, \ldots, 0)
\]

where the integer \(e_i\) is in the range \([0, m_i]\). Note that the non-zero number \((0, 0, \ldots, e_i, 0, \ldots, 0) = 0 \mod m_i\) for all \(j \neq i\). The error value is \(\bar{X}_i - X\) is denoted by \(E_{i,p}\). Then

\[
E_{i,p} = \bar{X}_i - X = (0, 0, \ldots, e_i, 0, \ldots, 0) = p \cdot M_i
\]

where \(M_i = \prod_{j=1,j \neq i}^{n} m_j\) and the non-zero integer \(p\) lies in the interval \([0, m_i]\). \(p = 0\) means that there is no error.

To calculate \(E_{i,p}\), both \(p\) and the error location \(i\) have to be computed simultaneously. The values, \(e_{i+1}\) and \(s_{i+2}\) introduced in [3] are related to \(E_{i,p}\). When a single error occurs in any one of the non-redundant residue digits, the redundant residue digits calculated using base extension differ from the original redundant residue digits. Hence,

\[
s_{i+1} = |\bar{x}_{i+1} - x_{i+1}|_{m_{i+1}} = |\bar{X}_i|_{m_{i+1}} - |X|_{m_{i+1}} \mod m_{i+1}
\]

\[
= |\bar{X}_i - X|_{m_{i+1}} = |E_{i,p}|_{m_{i+1}}
\]

Similarly, \(s_{i+2}\) can be shown to be equal to \(|E_{i,p}|_{m_{i+2}}\). Using Equation 2, \(s_{i+1} = p \cdot M_i \mod m_{i+1}\), and then \(p = \left|E_{i,p+1}|_{m_{i+1}}\right| \mod m_i\). Hence, the calculation of \(p\) needs the division by \(M_i\) in both modulo \(m_{i+1}\) and modulo \(m_i\). These divisions require that both of the redundant moduli be prime numbers because a multiplicative inverse of \(M_i\) will be used to perform the modulo division by \(M_i\). Since the mixed radix converter also requires the multiplicative inverse of each modulus (for \(m_i\) division), it is understood that once a mixed radix converter is used for base extension, the prime modulus condition must always be exercised. Based on the results \(s_{i+1} = |p \cdot M_i|_{m_{i+1}}\) and \(s_{i+2} = |p \cdot M_i|_{m_{i+2}}\), we establish the following two theorems to calculate the error vector \(E_{i,p}\). Depending on the error \(\bar{X}_i > X\) and \(\bar{X}_i < X\), \(E_{i,p}\) may be positive or negative. These two cases are treated separately in the remainder of this paper.

**Case 1** : \(E_{i,p} > 0\) (Positive)

The following theorem provides a means for finding the value of \(p\) and the error location \((i\)th digit).

**Theorem 1** : For \(E_{i,p} > 0\) if one of the non-redundant residue digits \((x_i)\) is faulty, then

\[
|E_{i,p+1}|_{m_{i+1}} = |E_{i,p+2}|_{m_{i+2}} = \frac{p}{m_i} \text{ if and only if } j = i,
\]

where \(|E_{i,p+1}|_{m_{i+1}} = s_{i+1}\) and \(|E_{i,p+2}|_{m_{i+2}} = s_{i+2}\).

**Proof** : (The proof of this theorem is given in Appendix A.)

The following example is used to illustrate the use of Theorem 1 to calculate the value of \(p\).

**Example 1** Consider the number \(X = 10\) represented by the moduli set, \((m_1, m_2, m_3, m_4, m_5, m_6) = (3, 7, 11, 13, 17, 19)\), where \(m_5\) and \(m_6\) are redundant moduli. The residue representation of \(X\) is given by: \(X = 10 = (1, 3, 10, 10, 10, 10)\). If an error occurs in \(x_2\) digit \((i = 2)\) such that it changes from 3 to 2, then the erroneous number \(\bar{X}_2\) generated by the non-redundant residue digits is given by: \(\bar{X}_2 = (1, 2, 10, 10, 10, 10)\). Now, \(|E_{2,p}|_{m_6} = s_4 = |E_{2,p}|_{m_6} = \left|\frac{6 - 10}{17}\right| = 12\) and similarly, \(|E_{2,p}|_{m_5} = s_5 = 14\). The value of \(M_2 = \prod_{j=1,j \neq 2}^{6} m_j = m_1 \cdot m_3 \cdot m_4 = 429\). Then,

\[
|E_{2,p}|_{m_5} = \frac{12}{14} = \frac{12}{14} = 3.\text{ Similarly,}
\]

\[
|E_{2,p}|_{m_6} = \frac{14}{19} = \frac{14}{19} = 3.\text{ Hence } p = 3.\text{ This can also be verified from the equality } 3 \times 429 = 1287 = \bar{X}_2 - X.
\]

Considering \(j = 1\) \((j \neq i)\), \(M_1 = 7 \times 11 	imes 13 = 1001\). Now, \(|E_{1,p}|_{m_6} = s_6 = 11\) and \(|E_{1,p}|_{m_5} = \frac{14}{19} \neq \frac{14}{19} = \frac{14}{19} = 3\). All possible error values due to an error in \(x_2\) are given by: \(|E_{2,p}| = p \cdot 429\), where \(m_2 < p < m_1\), and \(p \neq 0\).

**Note** that for \(E_{i,p} > 0\), \(p\) lies in the range \([1, m_1 - 1]\), and hence \(|p|_{m_1} = |p|_{m_1}\). In the above example \(|p|_{m_1} = |3|_{17} = 3\), and \(|p|_{m_1} = |3|_{19} = 3\).

**Using Theorem 1** the error correction for the positive \(E_{i,p}\) case is done based on the following procedure.

**PROCEDURE 1**

1. **INITIALIZE** \(i = 1\).

2. Calculate \(\frac{|E_{i,p}|_{m_{i+1}}}{|M_i|_{m_{i+1}}} m_{i+1}\) and \(\frac{|E_{i,p}|_{m_{i+2}}}{|M_i|_{m_{i+2}}} m_{i+2}\).

3. Compare the two values in Step 2. If they are EQUAL, THEN \(p = \frac{|E_{i,p}|_{m_{i+1}}}{|M_i|_{m_{i+1}}} m_{i+1}\), and GO TO Step 5, ELSE \(i = i + 1\).

4. IF \(i = l + 1\), THEN there is no error, and STOP. ELSE GO TO Step 2.

5. Calculate \(E_{i,p} = p \cdot M_i\).

6. Subtract \(E_{i,p}\) from \(\bar{X}_i\) to get the correct number \(X\).

**Case 2** : \(E_{i,p} < 0\) (Negative)

In this case Theorem 1 is not valid. The following theorem must be used to find the value of \(p\) and the error location.

**Theorem 2** : For \(E_{i,p} < 0\), if an error occurs in one of the non-redundant residue digits \((x_i), 1 \leq i \leq 5\), then

\[
\left|\frac{|E_{i,p}|_{m_{i+1}}}{|M_i|_{m_{i+1}}} m_{i+1}\right| m_{i+1} + m_j = \left|\frac{|E_{i,p}|_{m_{i+2}}}{|M_i|_{m_{i+2}}} m_{i+2}\right| m_{i+2} + m_j = \frac{p}{m_i} \text{ if and only if } j = i.
\]

**Proof** : (The proof of this theorem is similar to that for Theorem 1 and is omitted here.)

To obtain the value of \(p\) we use the same procedure (Procedure 1) as given previously with Step 2 changed appropriately. \(p\) may be treated same as \(p\), but \(p\) is used here to indicate a negative value of \(E_{i,p}\). An example to illustrate the use of Theorem 2 in calculating the value of \(p\) is given below.
Example 2 Consider the same modulus set as in Example 1. Let $X = 1297 = (1, 2, 10, 5, 5)$. Assume that an error occurs in the $x_2$ digit $(i = 2)$, and let $x_2 = 3$. Then, $X_2 = (1, 3, 10, 10, 10) = 10$. Thus, $X_2 = X\cdot M$, where $M = \{1, 3, 10\}$. Similarly,
\[
\left| E_{x_2} \cdot M \right|_{m_2} + m_2 = 4. \text{ Hence } p = 4.
\]

Considering $j = 1, M_j = 1001$. Then,
\[
\left| E_{x_2} \cdot M \right|_{m_2} + m_1 = 15. \text{ Thus,}
\]
\[
\left| E_{x_2} \cdot M \right|_{m_2} + m_1 \neq \left| E_{x_2} \cdot M \right|_{m_2} + m_1.
\]

In a similar manner,
\[
\left| E_{x_2} \cdot M \right|_{m_2} + m_j = 0, \text{ where } j = 3 \text{ and } 4.
\]

In this case ($E_{x_2} < 0$), we have
\[
\left| E_{x_2} \cdot M \right|_{m_2} \neq \left| E_{x_2} \cdot M \right|_{m_2}.
\]

Thus, $E_{x_2} < 0$ and $H_{x_2} > 0$, and Theorem 1 is not valid.

Note that for $E_{x_2} < 0$ the error calculator computes $E_{x_2} \cdot M$ to be subtracted from $X_2$ under the modulo operation. From the above example, $p = 4$ and it is verified that $p \cdot M = 4 \times 429 = 1716 = X_2 = X^2 \cdot M$, where $M = \{1, 3, 10\}$. $\Box$

5 Hardware Implementation

A complete block diagram of the hardware setup of the error calculator is shown in Figure 2. It includes an adder stage for $E_{x_2} < 0$. The ROMs $A_i$ and $B_i$ (1 = 1, 2, . . . , l) perform the divisions in Step 2 of Procedure 1 concurrently. The P-select logic receives the outputs from the equality checkers ($M$ and $N$) and the two sets of outputs $(A_i$ or $B_i$, and $A_i$ or $B_i$) from the ROM $A_i$ (or ROM $B_i$) and the other from the adder. The outputs of P-select logic will be $p$ or $\bar{p}$. Pass logic is used for the implementation of P-selector logic [9,10]. The set of ROMs $C_i$, $i = 1, 2, . . . , l$ multiply their inputs by $M_i$. Thus only one of the ROMs $C_i$ for $i$th digit error) produces a non-zero output ($p = E_{x_1}$).

If the ROM size of this approach is compared with the size based on the method in [3,4], it is seen that there is a great savings (80%) in memory size over a ROM implementation, and about 50% savings over a PLA implementation. A comparison table is given in Table 1. This comparison is based on 3 non-redundant and 2 redundant moduli, each 5 bits wide, with the modulus set $(m_1, m_2, m_3, m_4, m_5) = (17, 19, 23, 29, 31)$. This gives an overall length of 12 bits ($17 \times 19 \times 23$).

Although all other additional combinational logic has been taken into account, the total area required by this look-up table approach is much smaller when compared with a single ROM implementation. However, this approach seems to produce more delays. The throughput rate can be increased by using pipelined structures like those used in the MRC itself [6].

6 Conclusions

For the single fault tolerant RNS-based systems the design of an error calculator has been developed. The computation of an error does not depend on a single look-up table, instead it is performed independently by each redundant modulus in parallel. The hardware for the error calculator is reduced significantly and is easily realizable even for large moduli.

Appendix

A Proof of Theorem 1

This proof is divided into two parts.

i) To prove the ‘if’ condition, consider $j = 1$.

\[
|E_{x_2} \cdot M|_{m_1} = |E_{x_2} \cdot M|_{m_2} = |E_{x_2} \cdot M|_{m_3} = |E_{x_2} \cdot M|_{m_4} = |E_{x_2} \cdot M|_{m_5}.
\]

Since $p > 0$, and $p < m_1 - 1 < m_4 + 1$, $p|E_{x_2} \cdot M|_{m_1} = p$. In a similar manner,

\[
|E_{x_2} \cdot M|_{m_2} = |E_{x_2} \cdot M|_{m_3} = |E_{x_2} \cdot M|_{m_4} = |E_{x_2} \cdot M|_{m_5} = p.
\]

Thus,

\[
|E_{x_2} \cdot M|_{m_2} = |E_{x_2} \cdot M|_{m_3} = |E_{x_2} \cdot M|_{m_4} = |E_{x_2} \cdot M|_{m_5}.
\]

ii) To prove the ‘only if’ condition, assume first that

\[
|E_{x_2} \cdot M|_{m_1} = |E_{x_2} \cdot M|_{m_2} = |E_{x_2} \cdot M|_{m_3} = |E_{x_2} \cdot M|_{m_4} = |E_{x_2} \cdot M|_{m_5} = A,
\]

and $j \neq i$ where $A$ is an integer less than $m_i + 1$. Now,

\[
|E_{x_2} \cdot M|_{m_1} = |E_{x_2} \cdot M|_{m_2} = |E_{x_2} \cdot M|_{m_3} = |E_{x_2} \cdot M|_{m_4} = |E_{x_2} \cdot M|_{m_5} = A
\]

Similarly,

\[
|E_{x_2} \cdot M|_{m_2} = |E_{x_2} \cdot M|_{m_3} = |E_{x_2} \cdot M|_{m_4} = |E_{x_2} \cdot M|_{m_5} = A
\]

Therefore,

\[
|E_{x_2} \cdot M|_{m_1} = |E_{x_2} \cdot M|_{m_2} = |E_{x_2} \cdot M|_{m_3} = |E_{x_2} \cdot M|_{m_4} = |E_{x_2} \cdot M|_{m_5} = A
\]

But

\[
p \cdot m_j \cdot \frac{1}{m_i} \left| E_{x_2} \cdot M \right|_{m_i} = A + Q_1 \cdot m_i
\]

where $Q_1$ is an integer.

\[
p \cdot m_j \cdot \frac{1}{m_i} \left| E_{x_2} \cdot M \right|_{m_i} = A + Q_2 \cdot m_i
\]

where $Q_2$ is an integer. Subtracting Equation 5 from Equation 4, we get:

\[
p \cdot m_j \cdot \frac{1}{m_i} \left| E_{x_2} \cdot M \right|_{m_i} = A + Q_2 \cdot m_i
\]

The multiplicative inverse of $m_i$ can be written as:

\[
\frac{1}{m_i} = \frac{1 + k_1 \cdot m_i}{m_i}
\]

and

\[
\frac{1}{m_i} = \frac{1 + k_2 \cdot m_i}{m_i}
\]

Rewriting Equation 6 using Equation 7,

\[
p \cdot m_j \cdot \frac{1 + k_1 \cdot m_i}{m_i} - Q_1 \cdot m_i + 1 = p \cdot m_j \cdot \frac{1 + k_2 \cdot m_i}{m_i} - Q_2 \cdot m_i + 1
\]

Subtracting $\frac{m_i}{m_i}$ from both sides and multiplying by $m_i$, we get:

\[
\frac{m_i}{m_i} \left( k_1 \cdot m_j \cdot p - Q_1 \cdot m_i \right) = \frac{m_i}{m_i} \left( k_2 \cdot m_j \cdot p - Q_2 \cdot m_i \right)
\]

Two cases are possible in Equation 9. They are: the quantities inside the parentheses on the left side and on the right side (1) are both zero, and (2) multiples of $m_i + 1$ and $m_i + 1$, respectively.

The former implies $k_1 \cdot m_j \cdot p = Q_1 \cdot m_i$. Using Equations 4 and 7, this becomes:

\[
\frac{1}{m_i} \cdot \left( \frac{1}{m_i} \left| E_{x_2} \cdot M \right|_{m_i} - 1 \right) \cdot m_j \cdot p = \frac{1}{m_i} \cdot \left( \frac{1}{m_i} \left| E_{x_2} \cdot M \right|_{m_i} \cdot m_j \cdot p - A \cdot m_i \right)
\]

This equality is satisfied only when $m_i \cdot p = A \cdot m_i$. Since $A < m_i + 1$, $p < m_i$, and $m_i$ and $m_i$ are relatively prime, it must be true that $m_i = m_i$ (i.e. $j = i$) and $A = p$ is the only solution that will satisfy the above
equality.

In the latter case, \((k_1 \cdot m_j - \frac{Q_j}{2} \cdot m_i) = \alpha \cdot m_{i+2}\), where \(\alpha\) is an arbitrary integer. Using Equations 4 & 7, replace \(k_1\) and \(Q_1\), then

\[
\frac{1}{m_{i+1}} \left( \left( \frac{1}{m_i} \cdot m_{i+1} \cdot m_i - m_i - 1 \right) \cdot \frac{m_j}{p} - \left( \frac{1}{m_i} \cdot m_i \cdot m_j \cdot p - A \right) \cdot m_j \right) = \alpha \cdot m_{i+2}.
\]

On simplification, we get:

\[-m_j \cdot p + A \cdot m_i = m_{i+1} \cdot m_{i+2} \cdot \alpha\]

Since \(A \leq m_{i+1} - 1\), it follows that \(A \cdot m_i - n_j \cdot p < A \cdot m_i < (m_{i+1} - 1) \cdot m_i < m_{i+1} \cdot m_{i+2} \cdot \alpha\). Thus \(-m_j \cdot p + A \cdot m_i \neq m_{i+1} \cdot m_{i+2} \cdot \alpha\).

Hence, the only case for which

\[\left| \frac{R_{j+1}}{M_j} \right|_{m_{i+1}} = \left| \frac{R_{i+1}}{M_{i+2}} \right|_{m_{i+2}} \text{ is } j = i.\]

Q.E.D.

References


Figure 1: A Concurrent Error Correction Scheme Using Base Extension

Figure 2: Error Calculator with \(i^{th}\) Digit Error: \(E_{i,M_j} \neq 0\)

<table>
<thead>
<tr>
<th>Single Stored Table</th>
<th>Our Method</th>
</tr>
</thead>
<tbody>
<tr>
<td>(ROM)</td>
<td>(PLA)</td>
</tr>
</tbody>
</table>

| General Form     | \(2^{2n} \times |M|\) | \(2n \times P^*\) \times |M| \| \| | \(2^n \times |M| \times |I|\) |
| Numerical Example| \(=2^{10} \times 12+171 \times 12+472\) \| \| | \(=2^{3} \times 30+2^{5} \times 36+2^{11}\) |

(Unit is the number of links in each memory)

\[P^* = \text{number of product terms} = 2 \sum_{i=1}^{n} (m_i - 1) + \sum_{i=1}^{n} (m_i - 1) + 1\]

Table 1: Hardware Comparison Table for Concurrent Error Calculator